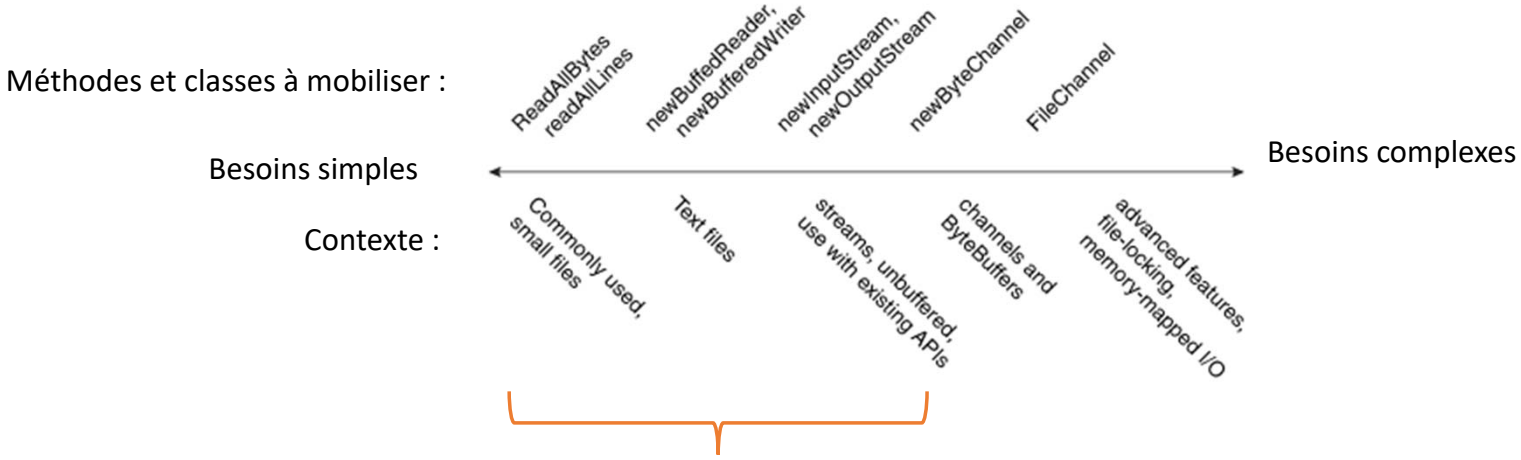
# Etape 6

- Flux avec java nio

# Gérer les flux avec NIO

Les *Files* et *Path.s* ne sont pas que des outils pour manipuler les fichiers ils peuvent aussi être utilisés afin d'écrire et lire dans des fichiers selon des modalités différentes des Stream de IO

Java présente ordonne cette api  NIO de la manière suivante (lien) :



- *readAllLines(),readString(), et lines* permetent de lire l'intégralité d'un fichier texte que l'on récupère en une seule String ou comme une liste/Stream de String

```
List<String> content = Files.readAllLines(fsource);
String allContent = Files.readString(fsource);
Stream<String> contentStream = Files.lines(fsource);
```

- Symétriquement on a les méthodes *Byte[] readAllBytes()* ou qui permettent en une seule instruction de lire l'intégralité d'un fichier binaire :

```
byte[]binContent = Files.readAllBytes(Paths.get("/img/logo.png"));
```

- Elles prennent aussi en paramètre le charset adequat (encodage des caractères du fichier)
- Pour être manipulables, les objets qui reçoivent les contenus doivent avoir une taille raisonnable => à réserver pour de petits fichiers

# Gérer les flux avec NIO

La méthode write() permet d'écrire du contenu dans fichier en une seule instruction. Elle possède plusieurs surcharges :

- Dont une première qui concerne les fichier binaires:

  Path write(Path path, byte[] bytes, OpenOption... options)

```
Files.write(fcible, binContent); // si binContent est un tableau de byte
```

- Et une deuxième concerne les fichier textes :

```
write(Path path, Iterable<? extends CharSequence> lines, Charset cs, OpenOption... options)
```

```
list<String> textContent;
… // peuplement de textContext
Files.write(fCible, textContent, Charset.defaultCharset());
```

## Faciliter la création de certains flux

Les méthodes *newBufferedReader()* et *newBufferedWriter()* sont des nouvelles méthodes de Files qui permettent d'obtenir directement des BufferedReader et BufferedWriter de IO que l'on obtenait auparavant avec un objet Reader vs Writer.

Les méthodes *newInputStream()* et *newOutputStream()* remplissent le même objectif (accourcir la création de certains flux ici les flux binaire en lecture et écriture vers des fichiers.

```
BufferedReader reader = Files.newBufferedReader(txtSrcePath,StandardCharsets.UTF_8);
BufferedWriter writer = Files.newBufferedWriter(txtCiblePath, StandardCharsets.UTF_8);
InputStream in = Files.newInputStream(binSrcPath);
OutputStream out = Files.newOutputStream(binCiblePath, StandardOpenOption.WRITE);
```